(19) Europäisches Patentamt
European Patent Office
Office uropéen des br vets

(11) EP 0 706 137 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
10.04.1996 Bulletin 1996/15

(51) Int Cl.[6]: **G06F 13/364**

(21) Application number: **95480127.0**

(22) Date of filing: **08.09.1995**

(84) Designated Contracting States:
AT BE CH DE ES FR GB IT LI NL SE

(30) Priority: **03.10.1994 US 317006**

(71) Applicant: **International Business Machines Corporation**
Armonk, N.Y. 10504 (US)

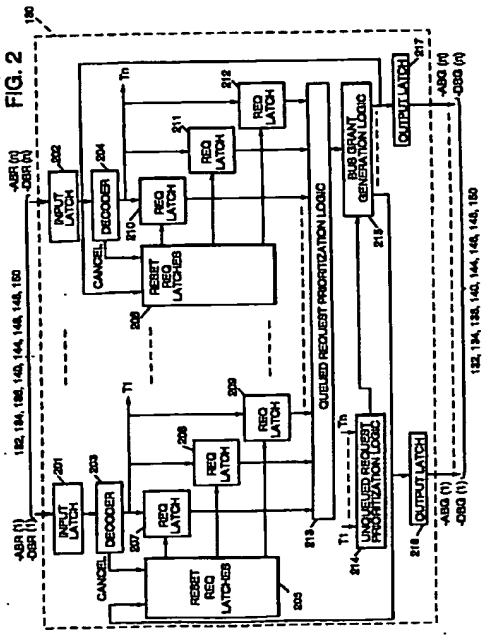
(72) Inventors:
- **Arimilli, Ravi Kumar**
  Round Rock, Texas (US)
- **Kaiser, John Michael**
  Cedar Park, Texas (US)

(74) Representative: **de Pena, Alain**
Compagnie IBM France
Département de Propriété Intellectuelle
F-06610 La Gaude (FR)

(54) Queued arbitration mechanism for data processing system

(57) A queued arbitration mechanism transfers all queued processor bus requests to a centralized system controller/arbiter in a descriptive and pipelined manner. Transferring these descriptive and pipelined bus requests to the system controller allows the system controller to optimize the system bus utilization via prioritization of all of the requested bus operations and pipelining appropriate bus grants. Intelligent bus request information is transferred to the system controller via encoding and serialization techniques.

FIG. 2



EP 0 706 137 A1

Description

CROSS REFERENCE TO RELATED APPLICATIONS

This application for patent is related to the following applications for patent filed concurrently herewith:

EFFICIENT ADDRESS TRANSFER TECHNIQUE FOR A DATA PROCESSING SYSTEM, serial number 08/AAA,AAA (HQ9-94-032);

DUAL LATENCY STATUS AND COHERENCY REPORTING FOR A MULTIPROCESSING SYSTEM, serial number 08/BBB,BBB (HQ9-94-033);

SYSTEM AND METHOD FOR DETERMINING SOURCE OF DATA IN A SYSTEM WITH INTERVENING CACHES, serial number 08/CCC,CCC (HQ9-94-034);

METHOD AND APPARATUS FOR REMOTE RETRY IN A DATA PROCESSING SYSTEM, serial number 08/DDD,DDD (AT9-94-163);

ARRAY CLOCKING METHOD AND APPARATUS FOR INPUT/OUTPUT SUBSYSTEMS, serial number 08/EEE,EEE (AT9-94-164);

DATA PROCESSING SYSTEM HAVING DEMAND BASED WRITE THROUGH CACHE WITH ENFORCED ORDERING, serial number 08/FFF,FFF (AT9-94-157);

COHERENCY AND SYNCHRONIZATION MECHANISMS FOR I/O CHANNEL CONTROLLERS IN A DATA PROCESSING SYSTEM, serial number 08/GGG,GGG (AT9-94-162);

ALTERNATING DATA VALID CONTROL SIGNALS FOR HIGH PERFORMANCE DATA TRANSFER, serial number 08/HHH,HHH (HQ9-94-035);

LOW LATENCY ERROR REPORTING FOR HIGH PERFORMANCE BUS, serial number 08/III,III (HQ9-94-036).

Each of such cross-referenced applications are hereby incorporated by reference into this Application as though fully set forth herein.

TECHNICAL FIELD OF THE INVENTION

The present invention relates in general to data processing systems and, in particular, to a system and method for intelligent communication of bus requests and bus grants within a data processing system.

BACKGROUND OF THE INVENTION

Conventional data processing systems, especially multiprocessor systems, allocate access to the shared system bus coupling the various bus devices to system memory through a mechanism whereby individual bus devices each control access to the system bus. Typically, each bus device will queue it's individual bus requests for various operations internally. Then, each bus device makes the determination of which of the various operations it wishes to perform on the system bus by sending the appropriate corresponding bus request to the system controller. Thus, each individual bus device determines internally which of its bus requests has higher priority. The system controller is then required to arbitrate between the received bus requests from the individual bus devices.

One disadvantage of this arbitration mechanism is that a portion of the decision process for accessing the various resources coupled to the system bus is delegated to each of the bus devices. As a result, the system controller is only able to view a portion of all of the various requests from the individual bus devices, since each of the individual bus devices retains and queues a significant number of bus requests. Thus, there is a need in the art for a more efficient arbitration mechanism for granting access to the system bus.

SUMMARY OF THE INVENTION

It is an object of the present invention to centralize the decision-making process for granting access to the system

bus. In an attainment of this object, the present invention provides a mechanism f transferring all of the qu ued bus requests from th individual bus devices to the system controller, which has a centralized knowledge of th availability of all of the system resources coupled to the system bus.

Th system controll r samples th bus devices' requests on a cycle-by-cycle basis. The requests ar encoded, which allows each of th bus devices to precisely communicate to the system controller each of their internally "qu ued" operations. Quickly transferring these "descriptive and pipelined" bus requests from each of the bus devices to a centralized control point, allows the system controller to "optimize" the system bus utilization by prioritizing all of the requested bus operations and pipelining the appropriate bus grants.

One advantage of the present invention is that it provides an ability to transfer "intelligent" bus request information from each bus device to the system controller, and provides the ability to transfer multiple packets of bus requests information (via encoding and serialization techniques).

Another advantage of the present invention is that the bus requests are compact and can be issued in a pipelined manner and that bus grants may be pipelined to either the same bus device or different bus devices.

Yet another advantage of the present invention is that it supports latch-to-latch or non-latch-to-latch implementations. Those skilled in the art will appreciate the benefit of accommodating both implementations. (Latch-to-latch implementations allow higher system bus clock rates, while non-latch-to-latch implementations provides lower system bus latencies.)

Yet still another advantage of the present invention is that the queuing of descriptive bus requests allows the system controller to efficiently control, distribute, and allocate system bus resources.

And, yet still another advantage of the present invention is that the system controller may resolve system level multiprocessor problems such as deadlocks and livelocks. Unlike traditional arbitration techniques, the present invention bus does not require bus devices to adhere to any arbitration "fairness" protocols.

Another advantage of the present invention is that the bus devices may support speculative bus requests and the system controller may support speculative bus grants.

The foregoing has outlined rather broadly the features and technical advantages of the present invention in order that the detailed description of the invention that follows may be better understood. Additional features and advantages of the invention will be described hereinafter which form the subject of the claims of the invention.

BRIEF DESCRIPTION OF THE DRAWING

For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following descriptions taken in conjunction with the accompanying drawings, in which:

FIGURE 1 illustrates a block diagram of a data processing system in accordance with the present invention;

FIGURE 2 illustrates a block diagram of the system controller illustrated in FIGURE 1; and

FIGURE 3 illustrates an exemplary protocol for granting bus grants for bus requests from one of the bus devices illustrated in FIGURE 1.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT OF THE INVENTION

With the foregoing hardware in mind, it is possible to explain the process-related features of the present invention. To more clearly describe these features of the present invention, discussion of other conventional features is omitted as being apparent to those skilled in the art. It is assumed that those skilled in the art are familiar with a multiuser, multiprocessor operating system, and in particular with the requirements of such an operating system for memory management including virtual memory, processor scheduling, synchronization facilities for both processes and processors, message passing, ordinary device drivers, terminal and network support, system initialization, interrupt management, system call facilities, and administrative facilities.

Referring now to FIGURE 1, a data processing system which advantageously embodies the present invention will be described. Multiprocessor system 100 includes a number of processing units 102, 104, 106 operatively connected to a system bus 108. Also connected to the system bus 108 is a memory controller 110, which controls access to system memory 112, and I/O channel controllers 114, 116, and 118. Additionally, a high performance I/O device 120 may be connected to the system bus 108. Each of the system elements described 102-120, inclusive, operate under the control of system controller 130 which communicates with each unit connected to the system bus 108 by point to point lines such as 132 t processor 102, 134 t processor 104, 136 to processor 106, 140 to memory controll r 110, 144 to I/O channel controller 114, 146 to I/O channel controller 116, 148 to I/O chann l controll r 118, and 150 to high performance I/O device 120. Requests and grants of bus access are all controlled by system controll r 130.

I/O channel controller 114 controls and is connected t system I/O subsystem and nativ I/O subsyst m 160.

Each processor unit 102, 104, 106 may includ a processor and a cach storage device.

The present invention may be implemented with a clock synchronous system bus and separate address and data buses. Furthermor , as illustrated in FIGURE 1, bus requests and bus grants ar transferred, in a pref rred embodiment, point-to-point (unidirectionally r bidirectionally) between th bus devices and system controller 130. This unidirectional point-to-point topology is capable of supporting totally independent and concurrent address and data bus requests and address and data bus grants. Additionally, it provides for system scalability without affecting the request-to-grant speed, and is capable of supporting a "private" communication protocol between the various bus devices and system controller 130.

Some of the possible encoded bus requests may be as follows:

XBR	-ABR	-DBR	Code	Address Bus Request	Data Bus Request	Queued Request	Cancel Queued Requests	Request Priority	Typical Bus Operation
0	0	0	(A)	Yes	Yes	Yes	No	High Low	Store/Push Retried Store, Cast Out, Speculative Store
0	0	1	(B)	Yes	No	Yes	No	High Low	Load/Address Only Retried Load/Address Only, Speculative Load/Address Only
0	1	0	(C)	No	Yes	Yes	No	High	Load Reply/Intervention
0	1	1	NULL	No	No	No	No	-	NULL
1	0	0	(D)	Yes	Yes	No	Yes	High	Store/Push
1	0	1	(E)	Yes	No	No	Yes	High	Load/Address Only
1	1	0	(F)	No	Yes	No	Yes	High	Load Reply/Intervention
1	1	1	CNCL	No	No	No	Yes	-	CANCEL ALL REQUESTS

Possible bus grants may be encoded as f llows:

-ABG	-DBG	CODE	Description	Grant to which Bus Requests	Typical Bus Operation
0	0	(AD)	Address and Data Bus Grant	A or D	Store or Push or Cast Out
0	1	(AO)	Address Only Bus Grant	B or E	Load or Address Only
1	0	(DO)	Data Only Bus Grant	C or F	Load Reply or Intervention
1	1	(NG)	No Grant	--	--

In the above tables, XBR represents a control bit, while -ABR and -DBR represent address bus requests and data bus requests, respectively. -ABG and -DBG represent address bus grant and data bus grant, respectively. As may be noted in the above bus request table, a "1" in the XBR portion of the bus request code represents that a particular bus device is sending a request that is not to be queued and which cancels all previously queued requests from that particular bus device.

Note that the terms "Cast Out", "Store", "Push", "Load", "Address Only", "Load Reply", "Intervention", "Speculative Load", "Speculative Store", "Retried Store", "Load Reply", "Retried Load" are all terms for operations well-known in the art and are to be interpreted according to their traditional references in the art.

Referring next to FIGURE 2, there is illustrated a block diagram of system controller 130. System controller 130, as previously illustrated in FIGURE 1, is coupled to the various bus devices via lines 132, 134, 136, 140, 144, 146, 148 and 150. These lines carry the encoded bus requests and transmit the encoded bus grant information to and from the bus devices.

In the following discussion, reference will only be made to input latch 201, decoder 203, reset request latches 205 and request latches 207-209, which are coupled to processor 102 via connection 132. Components 202, 204, 206, 210, 211, 212 operate in a similar manner, and may be coupled to I/O channel controller 118 via connection 148.

When a bus device, such as processor 102 sends a bus request to system controller 130 via line 132, it is received by input latch 201. Latches 201, 202, 216 and 217 assist in implementing system controller 130 with the bus devices in system 100 within a latch-to-latch implementation.

As bus requests are sent from processor 102 to system controller 130, they are latched into input latch 201 and decoded by decoder 203. If the bus requests are to be queued requests, they are then latched in succession into request latches 207-209. In a preferred embodiment of the present invention, system controller 130 implements a 3-deep queue. Of course, system controller 130 could be designed by one skilled in the art to implement various other N-deep queues, where N > 0. Note that at the same time that processor 102 is sending queued requests to system controller 130 as described above, other bus devices, such as I/O channel controller 118 may also be sending bus requests, whether queued or not queued, to be latched into request latches 210-212.

Queued request prioritization logic 213 then observes all latched requests from all bus devices via latches 207-212, and prioritizes their requests to determine which are to be given bus grants first. For example, by reference to the bus request table above, high priority requests will be granted access to system bus 108 before low priority requests. Furthermore, logic 213 may be designed for a particular system 100 to grant bus 108 for a load bus request before a store bus request. One skilled in the art may easily implement any desired priorities for determining which requests are to receive granting of bus 108 and in what order for implementation within logic 213.

As logic 213 determines which queued request to grant the bus to next, it then signals bus grant generation logic 215 of which encoded grant to generate and to which bus device.

If decoder 203 receives one of the bus requests from the bus request table that requires previously queued requests to be cancelled, decoder 203 will signal reset request latch 205, which resets request latches 207-209, cancelling previous requests from processor 102. Decoder 203 also sends these unqueued requests to unqueued request prioritization logic 214, which also performs a prioritization process between the unqueued requests received by logic 214. Logic 214, upon determining which of the unqueued requests is to be granted access to system bus 108 next, signals logic 215 of such a decision. Again, one skilled in the art may easily implement any desired priorities for determining which unqueued requests are to receive granting of bus 108. Logic 215 receives a prioritized queued request and a prioritized unqueued request, and determines which of these is to be granted access to system bus 108 next. Generally, since unqueued requests have a high priority, they will be granted access to system bus 108 before queued requests.

Bus grant generation logic 215 generates the encoded bus grants illustrated in the table above. These bus grants are latched out of system controller 130 by output latches 216-217. Thus, if logic 213 determines that a queued request from processor 102 is to receive the next bus grant, bus grant gen ration logic 215 will produce the appropriate encoded grant, which will be latched from latch 216 to processor 102, which will th n utiliz syst m bus 108 in th requested

manner.

Also, the encoded grant from bus grant generation logic 215 will b used by reset request latches 205 to reset th appropriate queued request latch 207, 208, or 209.

R ferring next to FIGURE 3, th re is illustrated an xemplary protocol of bus requests and bus grants for one of th bus devices, such as processor 102. The bus requests ar pipelined and sent from processor 102 via lin 132 to system controller 130 as indicated. Through the above-described process, system controller produces the pipeline of bus grants as indicated in FIGURE 3. Note the numerous "no grants" ("NGs") within the pipeline of grants, which may indicate that one or more of the other bus devices in system 100 is currently being served by system controller 130. The "null" notations indicate that processor 102 is currently not transmitting a bus request.

Since a queued bus request may have previously been transmitted by processor 102, a "null" request does not imply that processor 102 does not need access to the system bus 108.

The example shows that processor 102 is first in need of a store or push bus operation (noted by code A) and is next in need of a load or address only operation (noted by code B). At some time later, system controller 130 grants the address and data bus for the code A requested operation and then later grants the address bus only in response to the code B request.

The example bus request pipeline also indicates a serialization technique whereby two consecutive encoded requests from a particular bus device indicate to system controller 130 that the bus request is a low priority request. Such a low priority request may be in response to a previously "retried" bus operation from one of the other bus devices. Retries on the system bus 108 often result in prolonged livelocks and maybe even a deadlock. A deadlock may be defined as an infinite livelock. A livelock may be defined as a condition on the system bus 108 in which a bus device "A" retries an operation by bus device "B" and bus device "B" retries an operation by bus device "A" and this cyclical pattern continues until another condition "alters" this pattern. Livelock conditions are well known in the art. Livelock conditions severely degrade system performance due to the inefficient usage of the system bus resources. Therefore, it is advantageous to differentiate a bus request from a previously retried bus request. Furthermore, retried bus requests often get retried again due to "busy" system resources. Thus, it is also advantageous to have these retried bus requests contain a low priority in order to more efficiently utilize the system bus resources. It may further be preferable, then, to grant access to the bus for low priority requests in a randomized fashion and high priority requests in a prioritized fashion. The random generation of grant to low priority requests avoids the cyclical system bus retries, thus avoiding livelocks and deadlocks. The prioritized generation of grant to high and low priority requests efficiently realizes the system bus bandwidth.

As indicated within FIGURE 3, the bus request encoded as a load reply or intervention, is given a data only bus grant before the address and data bus grant in response to the low priority requests (encoded with an A). This illustrates how system controller 130 granted access to bus 108 to a higher priority request instead of a low priority request.

Also illustrated is how the second bus request B is cancelled by bus request D, which as indicated in the bus request table is not intended to be a queued request, and which informs system controller 130 to cancel all previous requests from processor 102. Since bus request D is not queued inside controller 130, bus request D remains active until it receives a grant. Such a situation is decoded by decoder 203 and transferred to unqueued request prioritization logic 214, which informs bus grant generation logic 215 of the unqueued request. Furthermore, decoder 203 informs reset request latches 205 to cancel all previous requests within the queued request latches 207-209.

In the above bus request and bus grant tables, intervention refers to a situation where another bus device has snooped a bus request and has determined that it contains within its internal cache a "dirty," or modified version of the requested data. A mechanism is then set in motion whereby the requesting bus device is informed that data is to be received from the other bus device instead of system memory. A further discussion of "intervention" is supplied within cross-referenced United States Patent Application No. (HQ9-94-034), which is hereby incorporated by reference herein.

Note that the XBR bus request signal need not be implemented in all systems. For low cost systems, the XBR information may be configured to a specific value in system controller 130 for certain bus devices.

Note that there is capacity for other encoded requests and grants via serialization techniques or the addition of bus request and bus grant signals. Furthermore, other types of requests and other protocols may be designed into the system of the present invention as desired.

Although the present invention and its advantages have been described in detail, it should be understood that various changes, substitutions and alterations can be made herein without departing from the spirit and scope of the invention as defined by the appended claims.

Claims

1. A computer system, comprising:

a plurality of bus devices including one or more processors and one or more storage devices;

a system controller;

a bus architecture coupling said plurality of bus devices and said system controller;

first means adaptable for transferring each address and data bus request from said plurality of bus devices to said system controller as said each address and data bus request is generated by said plurality of bus devices; and

second means adaptable for transferring, from said system controller to said plurality of bus devices, responses to said each address and data bus request.

2. The computer system as recited in claim 1, wherein said each address and data bus request is queued within said system controller and is not queued in said plurality of bus devices, and wherein said first means adaptable for transferring further comprises:

means for pipelining bus requests from each one of said plurality of bus devices; and

wherein said second means adaptable for transferring further comprises:

means for pipelining bus grants to each one of said plurality of bus devices.

3. The computer system as recited in claim 1, wherein said second means adaptable for transferring further comprises:

means for reordering bus grants, characterized in that at least one of said each address and data bus request is a speculative bus request; and

at least one of said responses to said each address and data bus request is a speculative bus grant.

4. The computer system as recited in claim 1, wherein said each address and data bus request and said responses to said each address and data bus request are performed on a point-to-point basis between each of said plurality of bus devices and said system controller; and

said bus architecture including separate address and data buses and said storage device comprising a memory controller coupled to a memory device;

said system controller sampling said each address and data bus request every bus cycle.

5. The computer system as recited in claim 4, wherein one of said each address and data bus request cancels all previous bus requests from one of said plurality of bus devices generating said one of said each address and data bus request cancelling said all previous bus requests.

6. The computer system as recited in claim 4, wherein an issue of one of said each address and data bus request on at least two consecutive bus cycles is treated as a low priority bus request by said system controller, wherein said system controller may issue a bus grant in response to a bus request received subsequent to said issue of said one of said each address and data bus request on at least two consecutive bus cycles before issuing a bus grant in response to said issue of said one of said each address and data bus request on at least two consecutive bus cycles.

7. The computer system as recited in claim 1, wherein said responses are ordered by said system controller according to priority levels of said each address and data bus request and according to an availability of system resources; and wherein one or more of said one or more processors are virtual processors.

8. A method of arbitrating bus access in a multiprocessor system, said method comprising the steps of:

sampling bus requests from a plurality of bus devices within said multiprocessor system in a manner that relieves queuing of said bus requests in each of said plurality of bus devices;

responding to said bus requests from said plurality of bus devices.

9. The method as recited in claim 8, wherein said sampling is performed every bus cycle by a system controller coupled in a point-to-point manner said plurality of bus devices, further comprising the step of:

queuing said sampled bus requests in said system controller.

10. The method as recited in claim 9, further comprising the step of:

issuing said bus requests by said plurality of bus devices, said issuing step occurring before said sampling step for each of said bus requests; and

wherein said issuing step further comprises the step of:

encoding one of said bus requests to be a request either of a store operation, a load operation, load reply, retried store, bus grant for said load operation, bus grant for said operation.

FIG. 1

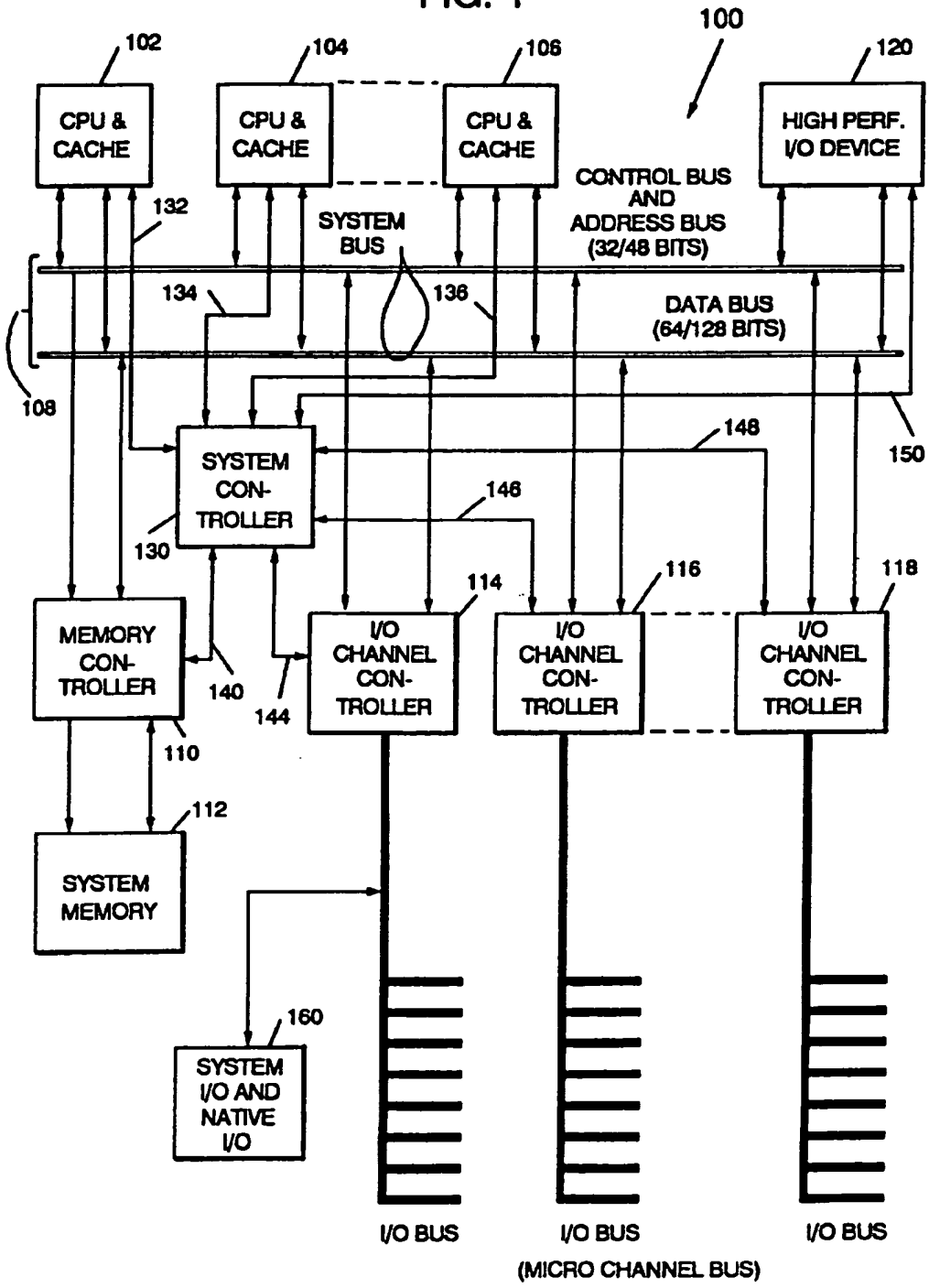

100
102
104
106
120
CPU & CACHE
CPU & CACHE
CPU & CACHE
HIGH PERF. I/O DEVICE
CONTROL BUS AND ADDRESS BUS (32/48 BITS)
SYSTEM BUS
132
134
136
DATA BUS (64/128 BITS)
108
148
150
SYSTEM CON-TROLLER
146
130
114
116
118
MEMORY CON-TROLLER
I/O CHANNEL CON-TROLLER
I/O CHANNEL CON-TROLLER
I/O CHANNEL CON-TROLLER
140
144
110
112
SYSTEM MEMORY
160
SYSTEM I/O AND NATIVE I/O
I/O BUS
I/O BUS
I/O BUS
(MICRO CHANNEL BUS)

FIG. 2

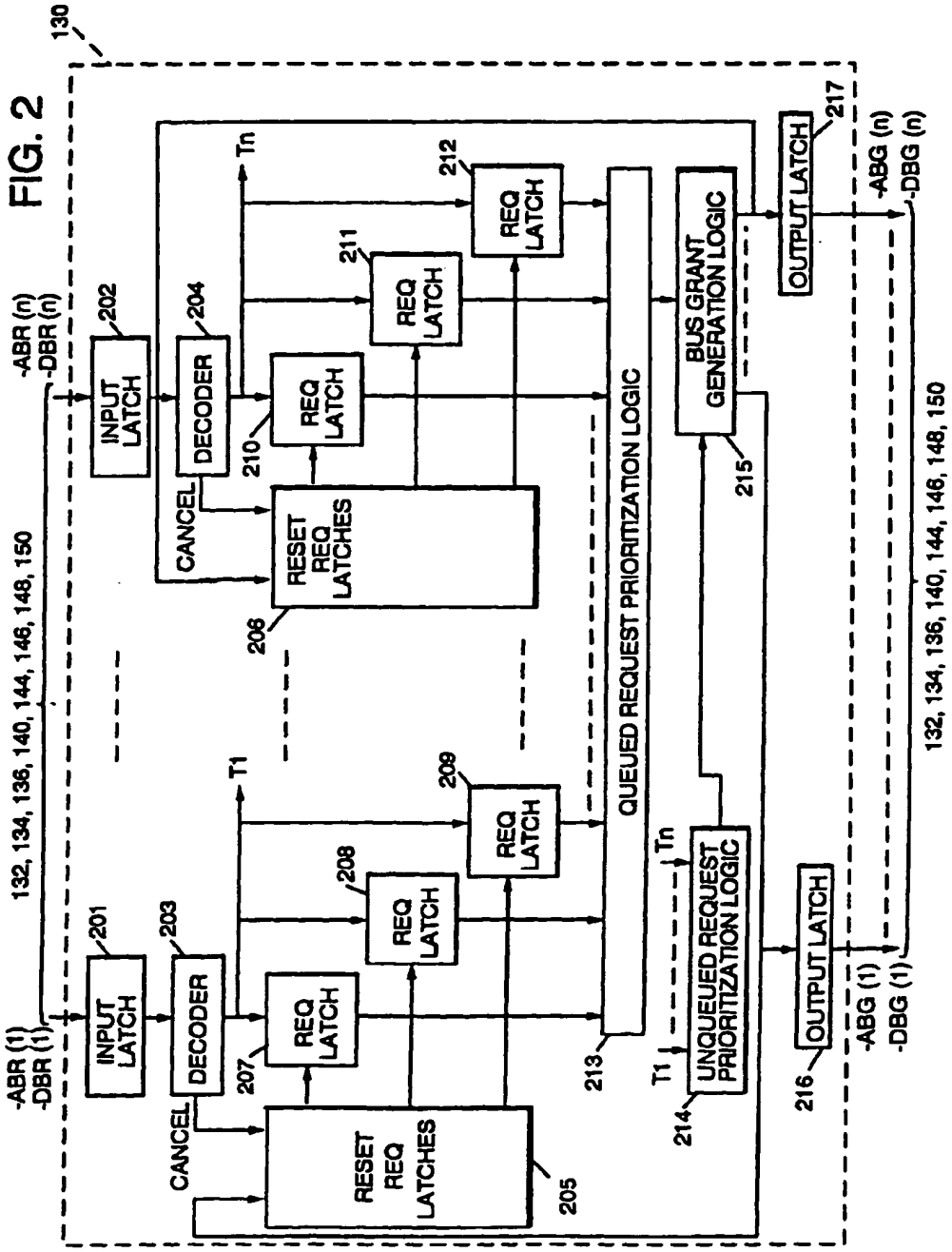
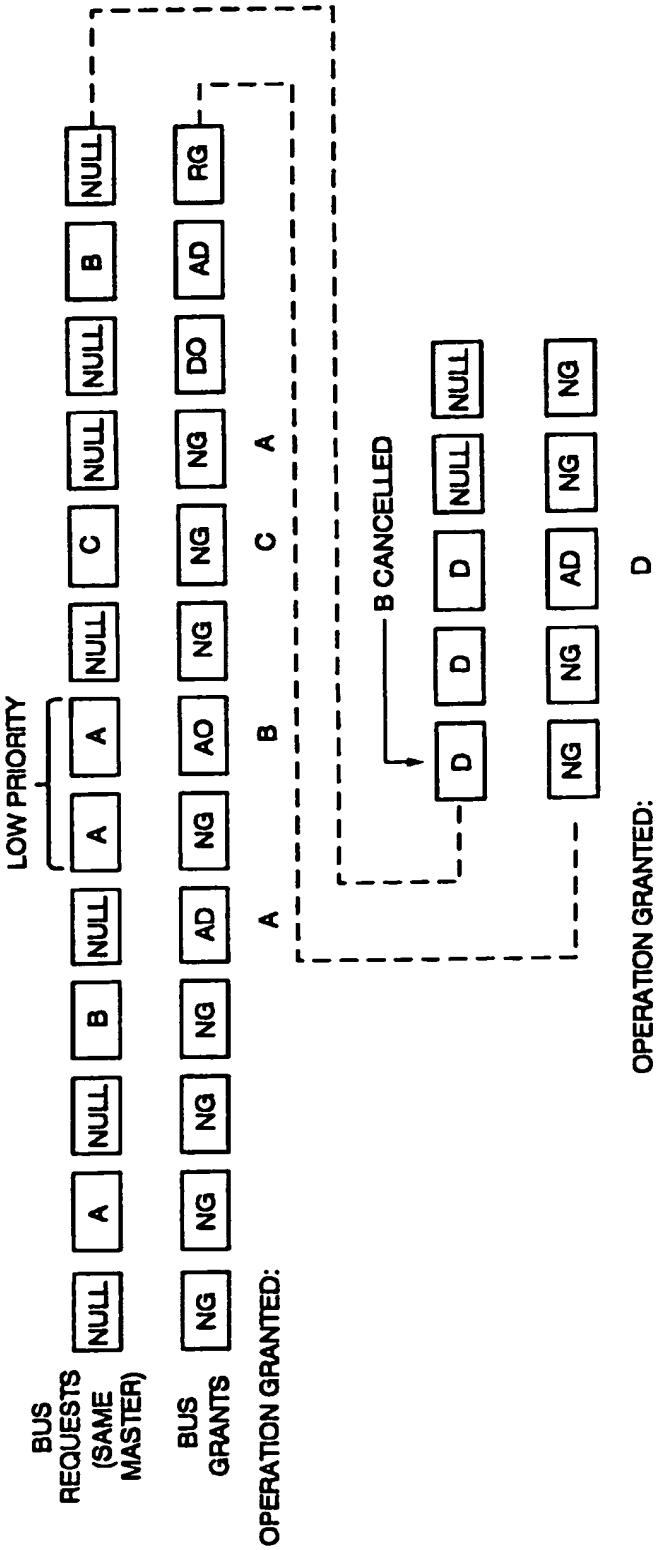

130
-ABR (n)
-DBR (n)
-ABR (1)
-DBR (1)
132, 134, 136, 140, 144, 146, 148, 150
INPUT LATCH
DECODER
CANCEL
RESET REQ LATCHES
REQ LATCH
QUEUED REQUEST PRIORITIZATION LOGIC
UNQUEUED REQUEST PRIORITIZATION LOGIC
BUS GRANT GENERATION LOGIC
OUTPUT LATCH
Tn
T1
-ABG (n)
-DBG (n)
-ABG (1)
-DBG (1)
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217

FIG. 3


BUS REQUESTS (SAME MASTER)
BUS GRANTS
OPERATION GRANTED:
LOW PRIORITY
NULL
NG
A
NULL
NG
B
NULL
AD
A
NG
NULL
AD
A
NG
C
NULL
NG
NULL
NG
B
NULL
DO
AD
RG
A
B
C
A
B CANCELLED
D
D
D
NULL
NULL
NG
NG
AD
NG
NG
D

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 95 48 0127

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	WO-A-91 20041 (SUPERCOMPUTER SYSTEMS LIMITED PARTNERSHIP) * page 3, line 12 - page 5, line 35 * * page 6, line 24 - line 35 * * page 10, line 10 - line 31 * * claims 7,8; figures 1,2 *	1,2,8,9	G06F13/364
X	EP-A-0 488 771 (XEROX CORPORATION) * column 6, line 22 - line 56 * * claims 1-5 *	1	
A	EP-A-0 347 763 (MODULAR COMPUTER SYSTEMS INCORPORATED) * column 2, line 27 - column 3, line 39 * * claims 1-3; figures 5,6 *	1-10	
A	US-A-5 303 382 (BUCH ET AL) * column 2, line 20 - column 3, line 33 * * claims 1-4; figures 1,2 *	1-10	
			TECHNICAL FIELDS SEARCHED (Int.Cl.6)
			G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	7 February 1996	McDonagh, F

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)